мІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

нАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «лЬВІВСЬКА ПОЛІТЕХНІКА»

ІНСТИТУТ КОМП’ЮТЕРНИХ НАУК ТА ІНОРМАЦІЙНИХ ТЕХНОЛОГІЙ

|  |
| --- |
|  |



ЗВІТ

Про виконання лабораторної роботи № 2

З дисципліни

«Спеціалізовані мови програмування»

на тему «Основи побудови об’єктно-орієнтованих додатків на Python»

Виконала:

студентка гр. ІТ-31

Проців Р.В.

Прийняв:

доц. каф. ІСМ

Щербак С.С.

Львів – 2023

**Мета роботи**: Розробка консольного калькулятора в об’єктно орієнтованому стилі з використанням класів

**Хід роботи**:

**Завдання 1**: Створення класу Calculator

Створіть клас Calculator, який буде служити основою для додатка калькулятора.

**Завдання 2**: Ініціалізація калькулятора

Реалізуйте метод \_\_init\_\_ у класі Calculator для ініціалізації необхідних атрибутів або змінних.

**Завдання 3**: Введення користувача

Перемістіть функціональність введення користувача в метод у межах класу Calculator. Метод повинен приймати введення для двох чисел і оператора.

**Завдання 4**: Перевірка оператора

Реалізуйте метод у класі Calculator, щоб перевірити, чи введений оператор є дійсним (тобто одним із +, -, \*, /). Відобразіть повідомлення про помилку, якщо він не є дійсним.

**Завдання 5:** Обчислення

Створіть метод у класі Calculator, який виконує обчислення на основі введення користувача (наприклад, додавання, віднімання, множення, ділення).

**Завдання 6:** Обробка помилок

Реалізуйте обробку помилок у межах класу Calculator для обробки ділення на нуль або інших потенційних помилок. Відобразіть відповідні повідомлення про помилку.

**Завдання 7:** Повторення обчислень

Додайте метод до класу Calculator, щоб запитати користувача, чи він хоче виконати ще одне обчислення. Якщо так, дозвольте йому ввести нові числа і оператор. Якщо ні, вийдіть з програми.

**Завдання 8:** Десяткові числа

Модифікуйте клас Calculator для обробки десяткових чисел (плаваюча кома) для більш точних обчислень.

**Завдання 9**: Додаткові операції

Розширте клас Calculator, щоб підтримувати додаткові операції, такі як піднесення до степеня (^), квадратний корінь (√) та залишок від ділення (%).

**Завдання 10**: Інтерфейс, зрозумілий для користувача

Покращте інтерфейс користувача у межах класу Calculator, надавши чіткі запити, повідомлення та форматування виводу для зручності читання.

**Програмний код**:

class Calculator:

def \_\_init\_\_(self):

"""

Initialize a Calculator object with a result attribute.

"""

self.result = None

def input\_numbers(self):

"""

Prompt the user to input two numbers and return them as a tuple.

Returns:

tuple: A tuple containing two numbers entered by the user.

"""

try:

num1 = float(input("Enter the first number: ")) # Prompt for the first number

num2 = float(input("Enter the second number: ")) # Prompt for the second number

return num1, num2

except ValueError:

print("Invalid input for numbers.") # Handle invalid input

return None

def input\_operator(self):

"""

Prompt the user to input an operator and return it.

Returns:

str: The operator entered by the user.

"""

operator = input("Enter operator (+, -, \*, /, ^, √, %): ") # Prompt for the operator

if operator not in ['+', '-', '\*', '/', '^', '√', '%']:

print("Invalid operator.") # Handle invalid operator

return None

return operator

def calculate(self):

"""

Perform a calculation based on user input and display the result.

"""

num1, num2 = self.input\_numbers()

operator = self.input\_operator()

if num1 is not None and num2 is not None and operator is not None:

try:

if operator == '+':

self.result = num1 + num2

elif operator == '-':

self.result = num1 - num2

elif operator == '\*':

self.result = num1 \* num2

elif operator == '/':

if num2 == 0:

print("Division by zero is not allowed.")

return

self.result = num1 / num2

elif operator == '^':

self.result = num1 \*\* num2

elif operator == '√':

self.result = num1 \*\* 0.5

elif operator == '%':

self.result = num1 % num2

print("Result: ", self.result)

except Exception as e:

print("Error during calculation:", str(e))

def run(self):

"""

Run the calculator, allowing the user to perform calculations repeatedly.

"""

while True:

self.calculate()

choice = input("Do you want to perform another calculation? (Yes/No): ")

if choice.lower() != 'yes':

break

class HistoryCalculator(Calculator):

def \_\_init\_\_(self):

"""

Initialize a HistoryCalculator object as a subclass of Calculator, with an additional history attribute.

"""

super().\_\_init\_\_()

self.history = []

def calculate(self):

"""

Perform a calculation, store the full operation and result in the history.

"""

num1, num2 = self.input\_numbers()

operator = self.input\_operator()

if num1 is not None and num2 is not None and operator is not None:

try:

result = None # Initialize the result variable

if operator == '+':

result = num1 + num2

elif operator == '-':

result = num1 - num2

elif operator == '\*':

result = num1 \* num2

elif operator == '/':

if num2 == 0:

print("Division by zero is not allowed.")

return

result = num1 / num2

elif operator == '^':

result = num1 \*\* num2

elif operator == '√':

result = num1 \*\* 0.5

elif operator == '%':

result = num1 % num2

if result is not None:

# Store the full operation and result in history

full\_operation = f"{num1} {operator} {num2} = {result}"

self.history.append(full\_operation)

print("Result: ", result)

self.result = result # Update the result attribute

except Exception as e:

print("Error during calculation:", str(e))

def show\_history(self):

"""

Display the history of calculations, including the full operations.

"""

print("Calculation history:")

for i, operation in enumerate(self.history, 1):

print(f"{i}: {operation}")

if \_\_name\_\_ == "\_\_main\_\_":

calc = HistoryCalculator()

calc.run()

calc.show\_history()

**GitHub Repository:** <https://github.com/fxxwol/python/blob/main/lab_2.py>

**Висновок:** в результаті виконання даної лабораторної роботи було перетворено консольний калькулятор у об'єктно-орієнтований калькулятор, використовуючи класи в Python. Цей проект допоміг вивчити концепції об'єктно-орієнтованого програмування та організацію, зберігаючи функціональність і інтерфейс користувача калькулятора.